FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING
DEGREE PROGRAMME IN WIRELESS COMMUNICATIONS ENGINEERING

MASTER'S THESIS

TRAFFIC CLASSIFICATION AND PREDICTION, AND FAST UPLINK GRANT ALLOCATION FOR MACHINE TYPE COMMUNICATIONS VIA SUPPORT VECTOR MACHINES AND LONG SHORT-TERM MEMORY

Author	Eslam Eldeeb
Supervisor	Hirley Alves
Second Examiner	Carlos Lima

December 2020

Eldeeb E. (2020) Traffic Classification and Prediction, and Fast Uplink Grant Allocation for Machine Type Communications via Support Vector Machines and Long Short-Term Memory. University of Oulu, Faculty of Information Technology and Electrical Engineering, Degree Programme in Wireless Communications Engineering. Master's Thesis, 43 p.

ABSTRACT

The current random access (RA) allocation techniques suffer from congestion and high signaling overhead while serving machine type communication (MTC) applications. Therefore, 3GPP has introduced the need to use fast uplink grant (FUG) allocation. This thesis proposes a novel FUG allocation based on support vector machine (SVM) and long short-term memory (LSTM). First, MTC devices are prioritized using SVM classifier. Second, LSTM architecture is used to predict activation time of each device. Both results are used to achieve an efficient resource scheduler in terms of the average latency and total throughput. Furthermore, a set of correction techniques is introduced to overcome the classification and prediction errors. The Coupled Markov Modulated Poisson Process (CMMPP) traffic model is applied to compare the proposed FUG allocation to other existing allocation techniques. In addition, an extended traffic model based CMMPP is used to evaluate the proposed algorithm in a more dense network. Our simulation results show the proposed model outperforms the existing RA allocation schemes by achieving the highest throughput and the lowest access delay when serving the target massive and critical MTC applications.

Keywords: Fast uplink grant, long short-term memory, machine type communications, resource allocation, support vector machines.

TABLE OF CONTENTS

FOREWORD

This thesis presents a resource allocation scheme for machine type communication devices. It was done at the Center of Wireless Communications (CWC) at the university of Oulu, Finland. I would like to thank Dr. Hirley Alves for giving me the chance to join the MTC research group and for his guidance during the work. I would like to thank Mohammad Shehab for his support all the time.

Oulu, 7th December, 2020

Eslam Eldeeb

LIST OF ABBREVIATIONS AND SYMBOLS

$\lambda_i[t]$	Poisson process rate
$s_n[t]$	Markov chain state
P	State transition matrix
π	State probability vector
P_C	State transition matrix of coordinated devices
P_U	State transition matrix of uncoordinated devices
$\theta_n[t]$	Space and time based samples
$\theta[t]$	Time based samples
δ_n	Space based samples
π_C	State probability vector of coordinated devices
π_U	State probability vector of uncoordinated devices
$f_\theta(t)$	Samples distribution in space and time
$f_{\delta_n}(x, y)$	Samples distribution in space
$f_{\theta_n}(x, y, t)$	Samples distribution in time
P_{NA}	Probability of no alarm
P_A	Probability of alarm
T	Start time of the background process
τ	Duration of the background process
μ_x	Mean of the background process in x-coordinates
μ_y	Mean of the background process in y-coordinates
σ_x	Variance of the background process in x-coordinates
σ_y	Variance of the background process in y-coordinates
$\vec{x_i}$	Training features
z_i	Training labels
$\vec{x}$	Decision hyberplane
ϕ	Transformation kernel
h_t	Neural Network hidden layer
y_t	Neural Network output
$E(w, b)$	Quadratic cost function
ΔE	Loss gradient
η	Learning rate
h_{t-1}	Previous short term memory
C_{t-1}	Previous long term memory
σ	Sigmoid function
f_t	Forget gate output
i_t	Learn gate output
$\tilde{C}_t$	Long term memory output from learn gate
o_t	Use gate output
T_{train}	Training time
$T_{predict}$	Prediction time
ΔT	Correction time
T_{margin}	Safety time
α	Exploration rate
C	Channel capacity
SNR	Signal to noise ratio

$\|h\|$	Communication channel
T_h	Hardware delay
T_q	Queuing delay
T_t	Transmission delay
T_a	Access delay

LTE	Long term evolution
MTC	Machine Type Communication
MTD	Machine Type Communication Device
NR	New radio
QoS	Quality of Service
eMBB	enhanced Mobile Broadband
mMTC	massive Machine Type Communication
URLLC	Ultra-reliable low-latency communication
RA	Random Access
BS	Base Station
UE	User Equipment
GF	Grant-Free
NOMA	Non-Orthogonal Multiple Access
FUG	Fast Uplink Grant
3GPP	3rd Generation Partnership Project
SVM	Support vector machine
ARMA	Autoregressive-moving-average
ARX	Autoregression with exogenous inputs
RNN	Recurrent neural network
ANN	Artificial neural network
LSTM	Long short-term memory
GRU	Gated recurrent unit
SSMM	Source Semi-Markov Model
PU	Periodic update
ED	Event-driven
PE	Payload exchange
MAB	multi-armed bandits
MMPP	Markov Modulated Poisson Processes
CMMPP	Coupled Markov Modulated Poisson Processes
CMAP	Coupled Markovian Arrival Process
M-CMMPP	M-background Processes Coupled Markov Modulated Poisson Processes
CR	Cognitive radio
SDR	Software-defined radio
FCC	Federal Communications Commission
PRACH	Physical Random Access Channel
eNodeB	E-UTRAN Node B
RAR	Random Access Response
DL-SCH	Downlink Shared Channel
TC-RNTI	Temporary Cell Radio-Network Temporary Identifier

UL-SCH	Uplink Shared Channel
HTC	Human type communication
ACB	Access class barring
EAB	Extended access barrier
RRC	Radio resource control
RBF	Radial basis function
MLPs	Multi-layer perceptrons
MSE	Minimum square error
GD	Gradient descent
SGD	Stochastic gradient descent
Rprop	Resilient backpropagation
AdaGrad	Adaptive gradient
Adadelta	Adaptive delta
BFGS	Broyden-Fletcher-Goldfarb-Shanno
L-BFGS	Limited-memory BFGS
CG	Conjugate gradient
Adam	Adaptive moment estimation
DT	Decision trees
RF	Random forests
NAB	Numenta Anomaly Benchmark
GB-RA	Grant based random access

1 INTRODUCTION

Cellular communications have experienced a paradigm shift in the recent years by introducing service modes dedicated to machine type communication (MTC) [1]. The new services enabled by the 5G new radio (NR) are classified into: enhanced mobile broadband (eMBB) - which is dedicated to conventional broadband connectivity; massive machine type communications (mMTC), and ultra-reliable low latency communications (URLLC) [2]. It is expected that many internet-of-things (IoT) applications will be supported such as autonomous vehicles, remote surgeries, and industrial IoT (IIoT) [3]. Due to the diversity among MTC applications, the quality-of-service (QoS) requirements vary. Therefore, understanding their heterogeneous traffic behaviour becomes an essential part in any communication system [4]. In fact, the design of efficient and robust models, which properly capture both communication requirements and radio channel dynamics, needs a deep understanding of all of the underlying network components. In this context, traffic modeling aims to capture the behaviour of the traffic using a probabilistic model that can be implemented easily and provide feasible means for efficient allocation of network resources [4].

1.1 Literature Review

According to the source semi-markov model (SSMM) model presented in [5], MTC traffic is classified into 3 different classes: *a) Periodic Update (PU)*, which describes a periodic traffic characterized by small number of short packets, *b) Event-Driven (ED)*, which describes a non-periodic traffic due to a certain random trigger at unknown time, and *c) Payload Exchange (PE)*, which describes a bursty traffic that usually comes after PU or ED traffic. Traffic models are classified into source traffic models and aggregated traffic models [6]. Source traffic models, which treat every MTD as a single separate entity, are very accurate, though become extremely complex when modeling highly dense networks. Aggregated traffic models treat all MTDs within the network as one entity by simply accumulating all the traffic to have one stream. IoT devices have become more complex than before, making aggregation of traffic less efficient. When compared to the source traffic approach, the aggregated traffic models are less complex at cost of lower accuracy. Authors in [7] designed Coupled Markov Modulated Poisson Process (CMMPP) so as to capture the traffic behaviour efficiently based on a master node, called a background process, that describes the event. Authors in [8] show that modeling CMMPP using multiple background processes is computationally expensive in time. They introduce Coupled Markovian Arrival Process (CMAP) model based on unicast and multicast distributions describing the regular traffic and traffic affected by events, respectively.

In the long-term evolution (LTE) systems, devices need to access the medium via random access (RA) procedures [9], thus to obtain a radio resource and transmit a packet, a user equipment (UE) undergoes a four-handshake procedure. This procedure suffers from high signaling overhead, which causes longer delays that prevent achieving the URLLC QoS requirements [10]. It also fails to meet one of the most challenging requirement of IIoT, namely the real-time performance due to high latency. Additionally, highly dense MTC deployments with hundreds of devices competing for meager resources will suffer from large number of preamble collisions, which cause longer delays and even

packets drop [11]. To handle such issues, many solutions have been implemented as complementary features to the existing RA resource allocation algorithm, but they fail to guarantee MTC demands [12]. Other researchers proposed other techniques like grant-free (GF) transmission, where each device chooses randomly a resource to transmit its packets without requests [13].

Although GF solutions overcome the exchange of messages that causes signaling overhead problem. However, for mMTC scenarios, where the number of devices is usually larger than the number of available resources, still suffer from large number of collisions and resulting longer delays. To address such problems, authors in [14] design a resource allocation algorithm using periodic resource pooling. Authors in [15] suggest a dynamic resource allocation scheme, that resolves the preamble collisions rather than avoiding it. Authors in [16] present a GF solution based on non-orthogonal multiple access (NOMA). Unfortunately, the proposed solutions suffer from undesired signaling overhead and collisions [17].

Recently, learning-based solutions have gained more attention to solve RA existing problems [18]. Fast uplink grant (FUG) was the first one introduced with the Release-13 of the 3rd Generation Partnership Project (3GPP) technical report in [19]. It is a learning-based resource allocation technique, where the base station (BS) grants resources to active devices once they are active based on predictive schemes. Moreover, the FUG reduces signaling overhead and completely remove collision. Authors in [20] present a traffic prediction framework of IoT devices, which is influenced by binary Markovian events. A distributed non-orthogonal multiple access solution based on reinforcement learning is used in [21], while authors in [22] present a sleeping multi-armed bandits (MAB) FUG solution. However, they focus only on achieving optimal resource allocation based on QoS requirements of each device by employing a source traffic predictor and capturing the traffic behavior efficiently .

1.2 Contribution

Our contribution extends the on CMMPP traffic model as in [7] to evaluate the performance of our proposed model compared to RA procedures. Our proposal enables the CMMPP model to perceive multiple background processes, which allows us to describe scenarios where the network is congested with large number of active devices and low number of resources. Different from [8], we introduce a model able to handle multiple background processes, called M-background processes CMMPP (M-CMMPP) model, without being computationally expensive. Furthermore, we propose a novel FUG algorithm based on 3 steps:

1. The device classification mechanism; the BS prioritizes the devices according to their QoS requirements. In order to classify the devices, we employ a large-margin classifier based on support vector machines (SVMs), which searches for the optimal decision boundary that is maximally far away from closest points of each class in the training set [23], [24].

2. The traffic prediction mechanism; it is typically a time-series problem. The BS monitors the activity behaviour of each MTD and forecasts which devices are active or inactive in a given transmission instant. Many classical approaches have been

developed to forecast those time-series problems such as autoregressive-moving-average (ARMA), classic autoregression with exogenous inputs (ARX), Support Vector Regression (linear and non-linear), and gradient boosting trees [25]. Recently, applications are becoming increasingly, while the samples interdependence is becoming nonlinear. Therefore, many researchers start focusing on machine learning and deep learning approaches to solve complex time-series problems. Recurrent neural network (RNN) is a special kind of artificial neural network (ANN), that depends on its internal memory to handle time-series data. The concept of RNN was first developed in [26] by David E. Rumelhart. It showed outstanding performance in different applications such as speech recognition, hand written recognition, and time-series problems [27]. Based on the RNN concept, many models have been proposed such as long-short term-memory (LSTM) in [28], and gated-recurrent unit (GRU) in [29]. LSTM is a special kind of RNN, that comprises 4 neural network layers, instead of just one, and acts as gates along with some point-wise operations [28]. These gates are trained to learn what to use from short and long past and what to forget. We apply LSTM for traffic prediction in our proposed FUG model. Moreover, we discuss the rationale for using LSTM and how it can be applied in real-time.

3. The resource allocation mechanism; the BS uses device classification and traffic prediction results to grant resources to devices which are worth. CMMPP and M-CMMPP models are introduced in the problem formulation, where we will apply the proposed FUG algorithm and compare it to existing resource allocation techniques.

1.3 Outline

The rest of the thesis is organized as follows. Section 2 presents the system model and problem formulation. Section 3 presents brief overview on existing resource allocation schemes. In Section 4, we introduce the details of the proposed FUG solution. In Section 5, we present the simulation results and discussion. Finally, discussion and summary are presented in Section 6 and Section 7, respectively.

2 SYSTEM MODEL AND PROBLEM FORMULATION

Consider a cellular network composed of a set $\mathcal{D}$ of static MTDs served by one BS with limited number of frequency resources. The total number of devices is $D = |\mathcal{D}|$, where $|.|$ returns the length of a vector , such that each device $d_i \in \mathcal{D}$ can be either: *a) Active* or *b) Silent*. When Active, devices can transmit either data packets with lower priority, or alarm packets with higher priority. Each device $d_i \in D$ has fixed coordinate locations x_i and y_i, which are known to the BS. In this work, for each one of the devices, we aim to first predict its state, namely silent or active, as well as the corresponding traffic priority. Thereafter, the serving BS schedules the set of active devices according to their priorities. We formulate the system model using an efficient traffic model called CMMPP [7]. Then, we extend the baseline model to account for more dense scenario using M-CMMPP model, which consists of several background CMMPP processes. Fig. 1 presents the system model.

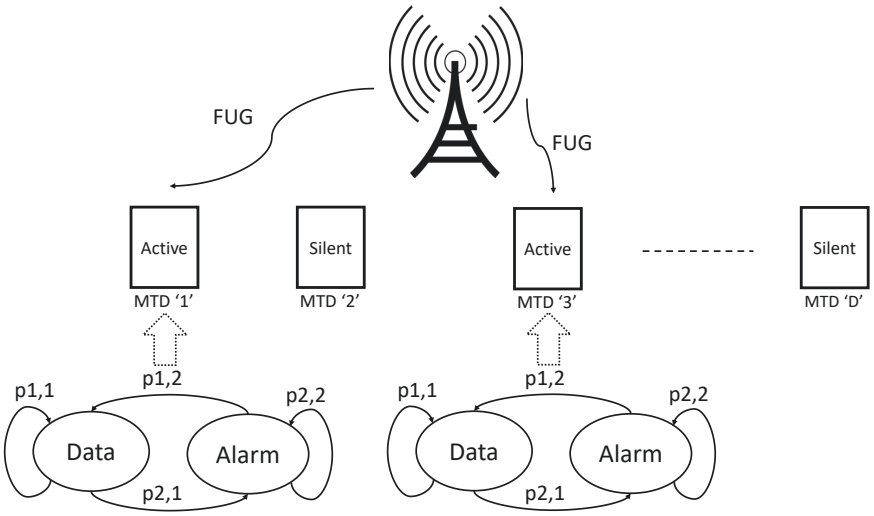


Figure 1: The considered system model. A set of D devices, which can be active or silent. Active devices can be in either in data or alarm state.

2.1 CMMPP Traffic Model

The Markov processes and Poisson processes are very popular traffic and queuing models [30], [31]. Recently, the Markov Modulated Poisson Processes (MMPP) are one of the recent theories that have been developed for traffic modeling scenarios, where a Poisson process with rate $\lambda_i[t]$ changes according to the state of Markov chains $s_n[t]$. Consider a two-state MMPP, where the data state describes regular packets transmission, while the alarm state describes longer and higher priority packets transmission. Each device transits between these two states according to the respective probabilities that based on the MMPP baseline model. Coupling Markov chains means that multiple chains mutually influence their transition probability matrices [7]. Given the state transition probabilities

we build a state transition matrix P_s, and while the state probability vector π is defined using the respective state probabilities as follows:

$$P_s = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,j} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,j} \\ \vdots & \vdots & \ddots & \vdots \\ p_{i,1} & p_{i,2} & \cdots & p_{i,j} \end{pmatrix}, \quad \pi = \begin{pmatrix} \pi_1 \\ \pi_2 \\ \vdots \end{pmatrix}. \tag{1}$$

In addition, tthe MMPP-based source traffic cam be simplified by considering only one background process to modulate all MTDs in case of a sudden event such as fire or high temperature. Thus, the background process influences all the MTDs in both space and time, but with different strengths according to their distances from the epicenter (position of the background process) and time of that background process. Moreover, it causes some devices to transition from state 1 (data) to state 2 (alarm), while others remain in their current state. The state probability matrix is composed of two matrices: the coordinated P_C and uncoordinated P_U matrices. The later describes the behaviour of the devices near to the epicenter and its main characteristic is to issue an alarm at a time (alarm state), then go back to the data state again. On the other hand, the later describes the behaviour of the devices away from the background process and its main characteristic is to remain at the data state and never switch to the alarm state. The background process generates samples $\theta_n[t]$, which are function of time and space for all devices. The state probability matrix is calculated as follows:

$$P_n[t] = \theta_n[t].P_C + (1 - \theta_n[t]).P_U, \tag{2}$$

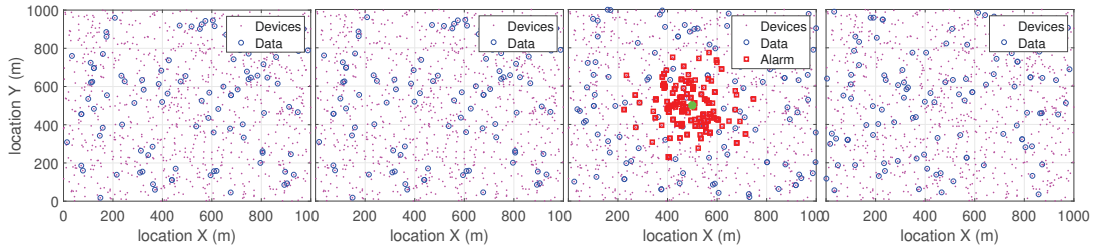
$$\theta_n[t] = \delta_n.\theta[t], \tag{3}$$

where $\theta[t] \in [0, 1]$ consists of samples, uniformly distributed over time, that describe how the devices are affected by the background process in time. δ_n follows standard normal distribution whose samples describe how the devices are affected by the background process in space. In addition, the mean and variance of δ_n describes the epicenter of the background process and how strong is the background process (high variance, means strong event and further devices are affected), respectively. Devices near to the epicenter of the background process are highly affected by the process and transit to state 2 (alarm). Multiplying $\theta[t]$ and δ_n results in $\theta_n[t]$, which describes how the devices are affected by the background process jointly in space and time. The same idea is repeated with the state probability vector $\pi_n[t]$, which will be composed of π_C and π_U. Therefore, θ and δ_n are distributed respectively as $\theta \sim \mathcal{U}(T, T + \tau)$ and $\delta_n \sim \mathrm{N}(\mu, \mathbf{\Sigma})$, where T is the start instant of the background process, τ is the duration of the background process, while μ and $\mathbf{\Sigma}$ represent the mean and the covariance matrix of the background process in x and y coordinates, respectively. Space and time distributions are independent. Therefore, the probability density function (PDF) of $\theta_n[t]$ is defined in (4), where $x, y \in \mathbb{R}_0^+$.

$$f_{\theta_n}(x, y, t) = \begin{cases} \frac{\frac{1}{\tau}}{2\pi\sigma_x\sigma_y} \exp\left(-\frac{1}{2}\left[\left(\frac{x-\mu_x}{\sigma_x}\right)^2 + \left(\frac{y-\mu_y}{\sigma_y}\right)^2\right]\right), T < t < T + \tau; \\ 0, \text{otherwise.} \end{cases} \tag{4}$$

Table 1: CMMPP Simulation Parameters.

Run Time	60 s
Time step (Instants for each second)	$\frac{1}{60}$
ΔT	1
Total Time Instants	1:3600
Number of Machines	1000
Number of States	2
P_C and P_U	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ and $\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$
λ_{Data}	$\frac{1}{3600}$
λ_{Alarm}	$\frac{1}{\delta T} = 1$
$T : T + \tau$	1700:1900
μ_x and μ_y	500
σ_x and σ_y	100
Intervals for the 4 States	[0 400; 1000 1400; 1600 2000; 2400 2800]



(a) Startup (b) Data (c) Alarm (d) Silent

Figure 2: CMMPP Results: 1000 MTD, 60 s runtime, 1000 m × 1000 m area, $\lambda_{Data} = \frac{1}{3600}$, and $\lambda_{Alarm} = 1$.

The CMMPP traffic model is simulated using the parameters in Table 1, while Fig. 2 describes the behavior of the MTDs in space and time. Fig. 2 shows (a) the Startup state where all the devices are likely to transmit data; (b) the Data state in which some devices transmit data at different instants and using different packet lengths; (c) the Alarm state where devices near to the epicenter transmit alarm signal in a correlated behaviour with large packet lengths due to the activation of the background process; and finally (d) the Silent state where the devices that transmitted alarm tend to be silent [7]. In addition, Fig. 3 illustrates the temporal evolution of the packets sizes for the MTDs.

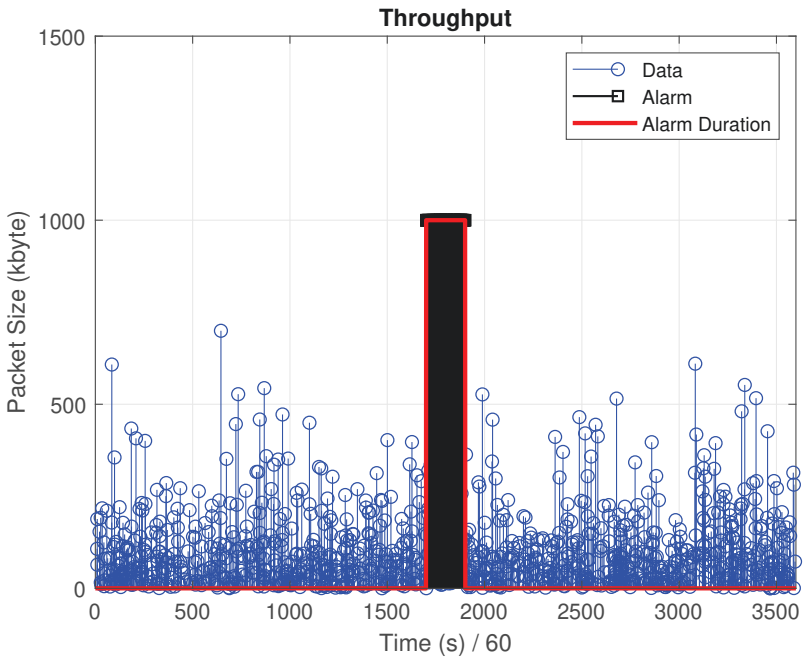


Figure 3: CMMPP traffic in time domain.

2.2 M-CMMPP Traffic Model

We introduced the CMMPP model based on one background process to represent a regular MTC application. Other applications can have more than one background process at a time causing a bursty scenario and network congestion [8]. Assume $M \geqslant 1$ background processes affect the transition of devices from the data to the alarm state as follows:

$$\theta_{n_m}[t] = \delta_{n_m}\theta_m[t], \quad m = 1, 2, 3, ..., M. \tag{5}$$

As the background processes are independent, We start by calculating the probability of no alarms (P_{NA}), which is a function of the probability of alarm at a given process (P_{m_A}):

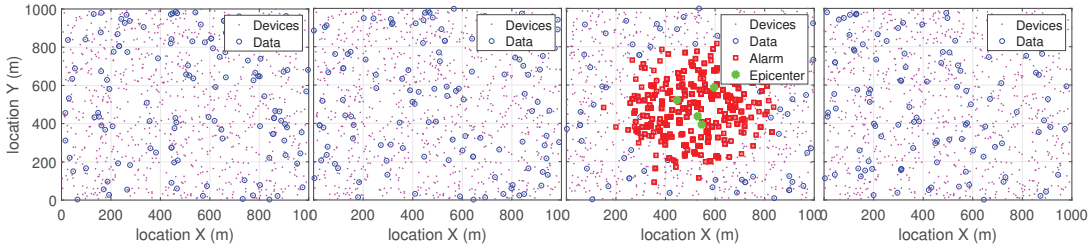
$$\begin{aligned} P_{NA} &= (1 - P_{1_A})(1 - P_{2_A}) \cdots (1 - P_{M_A}) \\ &= \prod_{m=1}^{M}(1 - P_{m_A}), \quad m = 1, 2, 3, ..., M, \end{aligned} \tag{6}$$

$$P_A = 1 - P_{NA} = 1 - \prod_{m=1}^{M}(1 - P_{m_A}), m = 1, 2, 3, ..., M. \tag{7}$$

The overall $\theta_n[t]$, which is the probability of having alarms due to M background processes, can be derived as follows:

$$\theta_n[t] = 1 - \prod_{m=1}^{M}(1 - (\delta_{n_m}.\theta_m[t])). \tag{8}$$

$$f_{\theta_n}(x,y,t) = \begin{cases} \frac{1}{\tau_m}\left[1 - \left(\Pi_{n=1}^{M}\left(1 - \frac{\exp\left(-\frac{1}{2}\left[\left(\frac{x-\mu_{n_x}}{\sigma_{n_x}}\right)^2 + \left(\frac{y-\mu_{n_y}}{\sigma_{n_y}}\right)^2\right]\right)}{2\pi\sigma_{n_x}\sigma_{n_y}}\right)\right)\right], T_m < t < T_m + \tau_m; \\ 0, \text{otherwise.} \end{cases} \tag{9}$$



(a) Startup (b) Data (c) Alarm (d) Silent

Figure 4: M-background Processes CMMPP Results: 1000 MTD, 60 s runtime, 1000 m × 1000 m area, $\lambda_{Data} = \frac{1}{3600}$, $\lambda_{Alarm} = 1$, 4 processes, and uniformly distributed processes.

Then, the overall $\theta_n[t]$ from (8) is used in (2) to form the transition probability matrix. Therefore, θ_m and δ_{n_m} are distributed respectively as $\theta_m \sim \mathsf{U}(T_m, T_m + \tau_m)$ and $\delta_{n_m} \sim \mathsf{N}(\mu_{\mathbf{m}}, \mathbf{\Sigma_m})$, where U is the uniform distribution, N is the normal distribution, $\mu_{\mathbf{m}}$ and $\mathbf{\Sigma_m}$ represent the mean and the covariance matrix of the background process m in x and y coordinates. Space and time distributions are independent. Therefore, the overall $\theta_n[t]$ is defined as in (9), where $x, y \in \mathbb{R}_0^+$, and $m = 1, 2, ..., M$.

Simulating the proposed M-background Processes CMMPP traffic model using four-back-ground processes shows similar results to the original CMMPP model, but with extremely larger number of data and alarm packets as shown in Fig. 4. This can cause network congestion, which is one of the most challenging problems in MTC applications while using current cellular network technologies. In addition, Fig. 5 illustrates MTD ready packets in time axis.

2.3 Conclusion

This chapter introduced an efficient traffic model to describe the behaviour of MTDs in a specific region. The traffic of 1000 devices was introduced assuming short data packets or long alarm packets due to an event. Moreover, we presented the M-background processes CMMPP to handle situation of extremely large number of packets in case of multiple events.

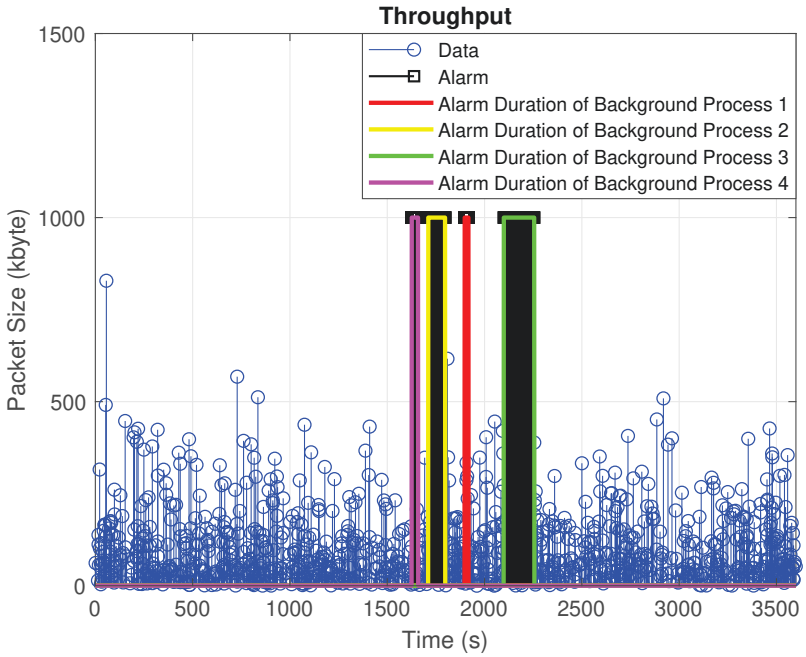


Figure 5: M-CMMPP results in time domain.

3 RESOURCE ALLOCATION ALGORITHMS FOR MTC APPLICATIONS

In this chapter, we present an overview on the current LTE RA allocation schemes. present-day random access schemes suffer from undesired signaling overhead, which fails to meet the requirements of MTC applications. Then, we present some alternative solutions to overcome the congestion in cellular networks. First, we consider RA-based scheme so as to limit the intrinsic overhead and latency. Afterwards, we present an alternative approach based on the GF scheme aiming to totally remove the signaling overhead. However, this approach still suffers from undesired problems deployment. This chapter mainly aims to identify the potential of machine learning-based solutions to solve the congestion problems for MTC applications in cellular networks. Fig. 6 shows a comparison between different resource allocation schemes in terms of signaling overhead.

3.1 Spectrum Sensing Resource Allocation

The frequency spectrum is the most valuable radio resource in wireless communication systems. There are some communication protocols (licensed and unlicensed), which are congested with users, while others are hardly accessed by users over time [32]. Cognitive radio (CR) approach improves the usage of spectrum resources by using, for example, software-defined radio (SDR) [33], [34]. According to [35], the Federal Communications Commission (FCC) is defining the CR as follows: "Cognitive radio: A radio or system that senses its operational electromagnetic environment and can dynamically and autonomously adjust its radio operating parameters to modify system operation, such as maximize throughput, mitigate interference, facilitate interoperability, access secondary markets". In fact, the cognitive radio is used to enhance the performance of communication systems. For example, it can be used to analyze the performance and availability of different networks, i.e. different bands, modulation, coding, and other communication aspects, then it can suggest the best choice for the user to establish communication [32]. A cognitive radio can sense the availability of spectrum, power needs, and channel properties. Despite having many features, we will focus only on the spectrum sensing techniques. Herein, a Spectrum sensing technique enables a device to identify the spectrum availability before transmitting. Implementing a resource allocation algorithm based on spectrum sensing techniques can eliminate collisions completely. In addition, it increases the overall throughput of the system. The concept of spectrum sensing is very wide and includes many algorithms, approaches, and complex hardware. Interested readers can refer to [32] for more information about CR, and to [36] and [37] for detailed discussion about spectrum sensing.

3.1.1 Challenges

Spectrum sensing algorithms overcome the collision problem by sensing the spectrum before transmission. However, one of the main problems of spectrum sensing algorithm is the large latency introduced from the sensing process of the availability of resources. Each device consumes large time trying to identify available resources. This large latency is

undesirable in MTC applications. In addition, spectrum sensing needs complex hardware [32].

3.2 Grant-Based Random Access Procedure

Random Access is a resource allocation approach, which is used in LTE networks [9]. There are two kinds of RA: contention-based and contention-free. In this section, we focus on former, which employs a 4-handshake procedure to allocate resources to a device as illustrated in Fig. 6 (a). The 4-handshake procedures is implemented through the following steps:

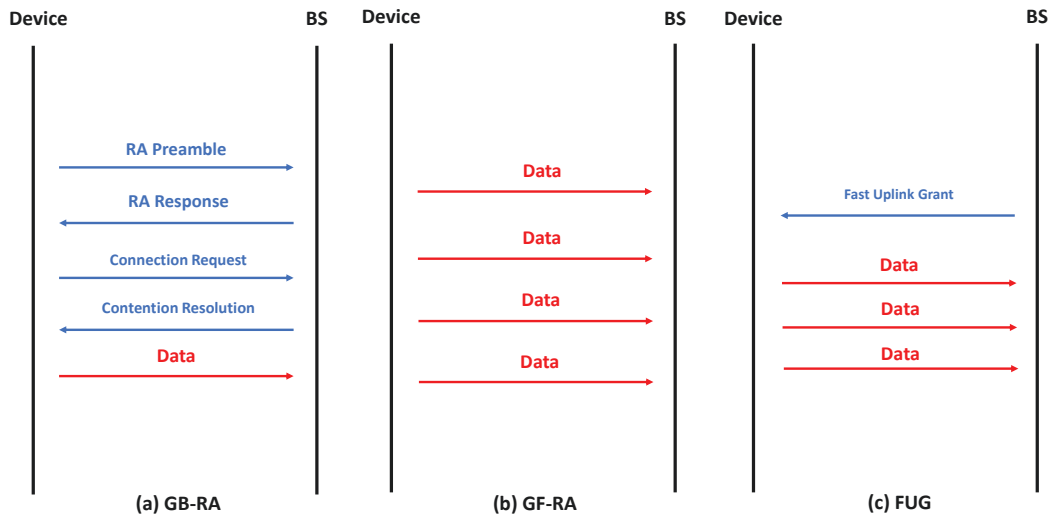


Figure 6: Grant-based random Access, grant-free random access, and fast uplink grant sequence diagrams.

1. There are preambles dedicated for initializing the RA. The UE randomly chooses one of the preambles and transmit RA preamble on Physical Random Access Channel (PRACH). This preamble is used by the eNodeB to estimate transmission timing.
2. Afterwards, the eNodeB transmits Random Access Response (RAR) on the Downlink Shared Channel (DL-SCH). This RAR corresponds to the selected preamble by the UE. The eNodeB transmits the resources allocated to UE in the message 3 of the 4-handshake procedures. In addition, it sends Temporary Cell Radio-Network Temporary Identifier (TC-RNTI) to be used in the upcoming communication steps between the UE and the eNodeB.
3. The third step is the connection request, where the UE transmits its identity to the eNodeB. It will use the the assigned resources from the eNodeB from message 2 and transmit its message through the Uplink Shared Channel (UL-SCH). The content of this message depends on the condition of the UE (active or silent) and whether it is known to that particular eNodeB or not.

4. The final message is the contention-resolution message. The eNodeB will compare the identity that is transmitted in message 3 with the one received in step 4. If the two identities match, it will establish the RA procedure successfully.

Each cell has 64 preambles dedicated for the RA procedure [9]. Those preambles are divided among 3 groups: group A, group B, and contention-free group. Group A is dedicated for high power devices, while group B is used by devices of less power. If the UE tries to perform an RA, it should choose a preamble randomly from one of the two groups: Group A and Group B. The selected group depends on the amount of information and the dedicated power used to transmit message 3 in the 4-handshake procedures. The contention-free group is used for handovers and other purposes.

If two UEs choose the same preamble at the same time, a collision occurs. This collision causes the two attempts to be dropped or one of them only. Let us see what happen in the second case, where one attempt successfully get the RA resource, while the other collides. Suppose two UEs access the same preamble at the same time. The eNodeB will receive both UEs message 1 and transmit message 2 for both UEs. Both UEs will transmit their identities through message 3, but when comparing the received identity from message 4, only one of them will have a matched identity, while the other after realizing that its attempt has failed will retry the RA attempt again at a later moment.

3.2.1 Challenges

Grant-based RA (GB-RA) resource allocation is very efficient and shows good results in LTE networks. However, GB-RA algorithm was designed to serve human type communication (HTC) devices and has been able to meet the requirements of HTC, which are not very strict and can handle some latencies while accessing the network. On the other hand, MTC devices have much more strict QoS requirements leading the GB-RA to not meet their requirements in terms of reliability and latency. The 4-handshake procedure itself adds undesired latency other than the transmission delay between transmitter and receiver. In addition, if a device collide during the RA, it will experience the full latency from the 4-handshake procedure, as it only realizes the collision at the last message. In order to handle such problems, many solutions have been implemented as complementary features in the existing RA resource allocation algorithm.

3.3 RAN Congestion in Cellular Networks and Existing Solutions

Due to the large number of MTC devices within one cell, large number of collisions occur. These collisions lead to high number of packet dropping and high latency intervals, which are not desirable for many MTC applications. To solve this congestion problem within cellular networks, many techniques have been developed recently. Next, we discuss a few existing techniques by comparing their main features and drawbacks.

(i) Push-based Scheme:

In this approach, devices randomly attempt RA without previous notice from the BS. There are different allocation techniques based on the push-based approach.

1. Back-off based scheme: After a collision, a device must wait for a back-off interval then re-transmit again [38]. This technique enhances the latency, but it still suffers

from some problems. Assume large packets are transmitted and they are using full capacity of the network. The long waiting time leads to packet drop. The main drawback of this technique is the high latency resulting from waiting and re-transmitting especially in critical application.

2. Access Class Barring (ACB) scheme: Different MTC devices are classified among different classes, each class takes an ACB parameter p_a which varies between 0 and 1, and an access baring timer [12]. Each device wants to access the BS sends a random number r, which varies between 0 and 1 as well. If $r < p_a$, the device starts RA procedure. If $r > p_a$, the device starts a timer which is proportion to the access baring timer, then tries to transmit again. This procedure performs well, but it suffers from several problems. If a device belongs to a class that has a very small ACB parameter, its opportunity to access the BS is very small. This leads to large amount of delays for those devices, and larger probability of packets drop. Moreover, this technique has a pre-step before the RA procedure, which increases the access latency. Many enhancements to this procedure have been proposed as discussed next:

    - Extended Access Barrier (EAB) scheme: When this mode is turned on, it prevents some devices, which belong to a certain access class, from accessing the network until it is switched off [39]. Despite the lower number of collision and higher success probability, it introduces high average delays.
    - Cooperative ACB scheme: As we know, each BS determines the ACB parameter p, which will be compared with the number generated by the device according to its class. However, in this approach more than one BS will cooperate to generate the ACB parameter p to reduce collision and delay at the same time [40].
    - Dynamic ACB scheme: Applying the concept of feedback, where the ACB parameter p is updated according to the previous time slots behavior in terms of congestion, collision, and delay [41].
    - Prioritized RA with dynamic ACB: It is similar to the dynamic ACB scheme, where previous time slots are monitored. In addition, instead of adjusting the ACB parameter p, the BS pre-allocates resources to certain classes and according to the feedback it changes the pre-allocated classes [42]. In the next chapter, will see that this approach is close to machine learning based solutions.

3. Dynamic Resource Allocation: This approach can be considered as a prediction-based technique, where the BS predicts the congestion and increases the number of available resources to serve more MTC devices [15]. The exhaustion of the available resources constitutes the main drawback here which may cause a high congestion situation, where BS does not have any available resources anymore.

4. Slotted Random Access: Each device device is assigned to a specific access slot and can only use that resource [12]. The main drawback is the large delay when the network suffers from high congestion.

5. Separation of RA Resources: In this approach, available resources are divided among HTC devices and MTC devices to avoid collision between them [12].

(ii) Pull-based Scheme:

It is also called Paging-based Scheme, where the BS first sends a paging message to the device in order to allow this device to start its RA attempt [12]. Hereafter, we discuss distinct implementations Pull-based approach.

1. Group-based RA Scheme: The main drawback of the Pull-based scheme is the large number of paging messages introduced by the BS for every device to allow this specific device to start RA attempt. One solution is to use group-based approach, where MTC devices are divided among some categories and all the devices classified in a certain category receive the paging message together [43]. For example, we can classify devices based on their QoS specifications or their geographic distribution.

It is worth mentioning Code-Expanded RA Scheme and Tree-based RA scheme [44], [45], [46].

Comparing the previous schemes, we will find a major trade-offs between achieving low delay, high energy efficiency, high probability of success, and guarantee the QoS for each application. We we can see that some techniques attain very high energy efficiency, but high delay on the same time like the slotted RA technique. Some techniques are good at certain criteria but cannot guarantee others. Hence, depending on the application requirement, the appropriate technique should be chosen. Furthermore, we need to develop new approaches that try to balances between all the criteria and achieve better results.

3.4 Grant-Free Approach

Grant-Free random access is proposed to overcome the problem of high signaling overhead in GB-RA schemes [47]. In GF, the prior requests done by the user are ignored, which allows the user to transmit information directly. As the signaling overhead is removed, the latency decreases. This approach is very efficient in periodic traffic scenarios and when the number of active devices is lower than the number of available resources. The needed control information, such as time, frequency, and power control settings are managed through Radio Resource Control (RRC) signaling before transmission. This approach decreases the latency and allow more critical devices to be served. However, when the number of active devices exceeds the number of available resources, it suffers from many collisions. These collisions cause the devices to wait and re-transmit again, which introduces high latency. In extreme cases, packets can be even dropped due to large waiting periods. In mMTC applications, the number of devices are more than the number of available devices, which results in longer latency and inefficient allocation. In addition, GF solutions require complex hardware at the receiver side and re-transmission of packets at collision at the transmitter side.

3.5 Summary and Conclusion

This section discussed the existing GB-RA resource allocation. We overview on GB-RA structure and the 4-handshake procedures. The limitations of current RA schemes in serving MTC applications were also presented. In addition, we showed some of the existing solutions in the literature and their drawbacks.

4 PROPOSED FAST UPLINK GRANT MODEL

The proposed FUG model is divided into 3 main parts: *i)* device classification, *ii)* traffic prediction, and *iii)* resource allocation, as shown in Fig. 7. Firstly, the BS predicts which devices, within a certain application under its coverage area, have higher priority compared to other devices and need to be served first. The second step is estimating the time, which each device needs to access the network. Finally, BS uses the results from the device classification and traffic prediction to allocate resources to selected devices at specific time instants.

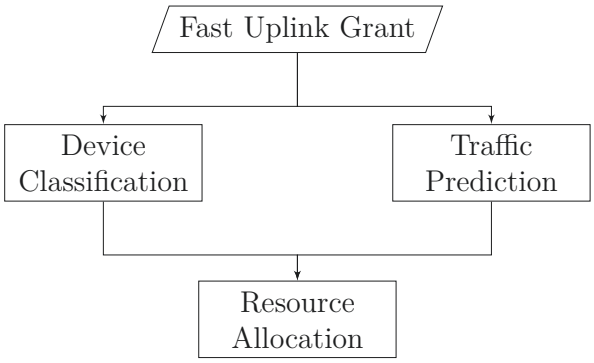


Figure 7: The proposed fast uplink grant model.

4.1 Device Classification Using SVM

According to the CMMPP model, the serving BS classifies devices into two groups, while devices in the first group transmit only data all the time, the second group transmit data and alarm. Devices that transmit alarm have higher priority than devices that transmit data. Specifically, our first step is a typical binary classification problem. Binary classification is a type of supervised learning, where a model is developed to classify some candidates among two classes. First step in device classification is collecting an appropriate dataset with desired features and labels, that reflects different scenarios of the application for a period of time. Features are the input data that the classifier should take into consideration and learn their pattern and how they are related to their corresponding labels. In MTDs classification, features are position coordinates of the devices, while labels are the data or alarm classes. Usually, binary classification problems are considered as finding the best decision boundary that separates the classes correctly.

The optimum decision boundary can be found using SVM algorithm. Define training points $P = (\vec{x}_i, z_i)$, where $\vec{x}_i$ are the features, and z_i are the labels $(-1, 1)$. Then, define the decision hyperplane as b and a normal vector $\vec{w}$ perpendicular to the hyperplane. Finally, define a point $\vec{x}$ to be on the hyperplane, so that:

$$\vec{w}^T \vec{x} = -b. \tag{10}$$

The SVM aims to maximize the width between the nearest features from one class to the other (support vectors of each class) [23]. Fortunately, this optimization problem

is convex, which can be solved using Lagrange multiplier theorem with any quadratic programming to find the solution:

$$L = \frac{1}{2}\|w\|^2 - \sum \alpha_i \left[z_i(\vec{w}.\vec{x}_i + b) - 1 \right], \tag{11}$$

where α_i is the Lagrange multiplier. Classify class 1 if:

$$\sum \alpha_i.y_i.\vec{x_i}.\vec{x} + b \leqslant 0. \tag{12}$$

By inserting features (device coordinates) and labels (class 1 or class 2) into the classifier, we find the optimal boundary between the two classes. However, this classifier is only applicable to feature points that are linearly separable. The CMMPP classes are non-linearly separable. Thus, to avoid this problem we use transformation kernels (ϕ) to map the feature points into higher dimensions, where they can be linearly separable:

$$\phi : \vec{x} \rightarrow \phi(\vec{x}). \tag{13}$$

The radial basis function (RBF) maps the data into an infinite dimensional Hilbert space [23]. It is very simple and fast to remap feature points using such kernel transformations; here, we employ the Gaussian RBF kernel given as

$$K(\vec{x}, \vec{x'}) = e^{-\frac{(\vec{x}-\vec{x'})^2}{2\sigma^2}}, \tag{14}$$

where $\vec{x}$ and $\vec{x'}$ are two features. The RBF SVM can not only extract the pattern of devices efficiently, but also classify them according to their priorities.

4.2 Traffic Prediction Using LSTM

After predicting the priorities of the devices, the serving BS needs to predict the activation and silent instants of these devices in order to implement an efficient resource scheduler. First, we explain how neural network works. Afterwards, we discuss briefly how recurrent neural networks and long short-term memory can be used in the proposed model.

4.2.1 Artificial Neural Networks

An ANN was mainly developed to be an alternative description of the logic gates such as NAND and NOT gates. ANN consists of layers of Perceptrons, called multi-layer perceptrons (MLPs). A perceptron is a function that accepts binary input and returns binary output. Sigmoid neurons were designed to accept non-binary inputs. Basic elements of ANN are: input features x_j, labels y_j, and a Sigmoid neuron, which has weights w_j and biases b_j. an ANN aims to set the appropriate weights and biases, which describe the relation between input features and the corresponding output labels. A simple ANN architecture consists of input features, a hidden layer (single or multiple neurons), and target output. Tupically, there are two kinds of weights connecting the layers: *(i)* w_{xh} are weights from input features to the hidden layer, and *(ii)* w_{hy} are weights from the hidden layer to the target output. This network is described as follows:

$$h = f\left(w_{xh}\, x\right), \tag{15}$$

$$y = w_{hy}\, h, \tag{16}$$

where the function f is the Sigmoid function and J is the length of the features:

$$f\left(w_{xh}\, x\right) = \frac{1}{1 + \exp\left(-\sum_{j=1}^{J} w_j\, x_j + b_j\right)}. \tag{17}$$

The most interesting property of Sigmoid functions is that small changes in weights and biases results in small changes in output, which can obtain the appropriate weights and biases that describe the model by slowly changing them. We notice that the output of a layer is used as an input to the next layer. This approach is called feedforward neural networks:

$$\Delta Output \approx \sum_{j=1}^{J} \frac{\partial Output}{\partial w_j}\, \Delta w_j + \frac{\partial Output}{\partial b_j}\, \Delta b_j. \tag{18}$$

Define the minimum square error (MSE), which is sometimes called quadratic cost function as follows:

$$E(w, b) = \frac{1}{2n} \sum \left\| y\left(x\right) - a \right\|^2, \tag{19}$$

where n is the number of inputs, $y\left(x\right)$ is the actual outputs, and a is the network output using the set of weights w and biases b. The main purpose of the network is to decrease the error between the actual outputs and the outputs of the network. This error is reduced by tuning the values of w and b until getting lower MSE. TIn order to minimize a function, we set its derivative equal to zero and get the weights and biases that achieve the minimum cost. However, this approach is only practical with a small number for low number of weights and biases. When considering large network architectures with huge number of weights, neurons and layers, the direct differentiation results in extremely complex equation with large number of variables. The gradient descent (GD) algorithm can be used to overcome this problem.

Consider the loss function as a valley and we want to reach the global minimum point, that is the point where specific combination of the variables gives the lowest possible MSE. We pick a random point and try to move away from this point into the global minimum of the function by updating the variables (weights and biases) slightly towards the global minimum. We employ (20) to slightly vary the loss function in a certain direction.

$$\begin{aligned} \Delta E &\approx \sum \frac{\partial E}{\partial w_j}\, \Delta w_j + \frac{\partial E}{\partial b_j}\, \Delta b_j \\ &\approx \frac{\partial E}{\partial w_1}\, \Delta w_1 + \frac{\partial E}{\partial w_2}\, \Delta w_2 + \frac{\partial E}{\partial w_3}\, \Delta w_3 + ...+ \\ &\frac{\partial E}{\partial b_1}\, \Delta b_1 + \frac{\partial E}{\partial b_2}\, \Delta b_2 + \frac{\partial E}{\partial b_3}\, \Delta b_3 + ..., \end{aligned} \tag{20}$$

where we need to ensure these small changes are actually leading in the direction of the minimum and ΔE is negative. The ∇E correpsonds to the gradient of the loss with respect to all parameters and vector w that contains all the parameters as follows:

$$\nabla E = \left(\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \frac{\partial E}{\partial w_3}, ..., \frac{\partial E}{\partial b_1}, \frac{\partial E}{\partial b_2}, \frac{\partial E}{\partial b_3}, ...\right)^T, \tag{21}$$

$$w = \left(w_1, w_2, w_3, ..., b_1, b_2, b_3, ...\right)^T, \tag{22}$$

where we can define $\Delta w = (\Delta w_1, \Delta w_2, \Delta w_3, ..., \Delta b_1, \Delta b_2, \Delta b_3, ...)^T$. T is the transpose operator. In addition, ΔE can be simplified as:

$$\Delta E \approx \nabla E \, \Delta w. \tag{23}$$

To ensure lower loss, the change in the parameters should be specified as a negative function of the gradient:

$$\Delta w = -\eta \, \nabla E, \tag{24}$$

$$\Delta E \approx -\eta \, \nabla E \, \nabla E = -\eta \, \|\nabla E\|^2 , \tag{25}$$

where η is a small positive number known as learning rate. We notice that $\|\nabla E\|^2$ and η are always positive, so (**??**) is always negative, which ensures that the gradient is going in the direction of the minimum. The next step is to update each weight and each bias as follows:

$$\begin{aligned} w_k^{'} &= w_k - \Delta w_k = w_k - \eta \, \nabla E \\ &= w_k - \eta \, \frac{\partial E}{\partial w_k}, \end{aligned} \tag{26}$$

$$\begin{aligned} b_l^{'} &= b_l - \Delta b_l = b_l - \eta \, \nabla E \\ &= b_l - \eta \, \frac{\partial E}{\partial w_l}. \end{aligned} \tag{27}$$

We repeatedly apply this equation to update the weights and bias that lead the cost function towards the global minimum point. Another challenging point in this gradient solution is choosing an appropriate learning rate η. In fact, by choosing a large learning rate, we could miss the global minimum point, whereas a very small learning rate could not reach the global minimum point. It is also worth identifying a few limitations of the standard gradient descent algorithms. For instance, when dealing with a cost function of adding input data (averaging individual costs) results in calculating the cost and gradient for each input. When considering thousands or even millions of input data, the resulting scenario becomes very complex and time-consuming. In such scenarios, the stochastic gradient descent (SGD) algorithm is an interesting alternative, where random training inputs are selected to calculate the gradients, then average the results of each of them. Applying optimization algorithms in neural network is an active area of research. Recently, many algorithms have been introduced such as the momentum, resilient backpropagation (Rprop), adaptive gradient (AdaGrad), adaptive delta (Adadelta), exponential decay learning rate, Quasi-Newton method including Broyden-Fletcher-Goldfarb-Shanno (BFGS) and Limited-memory BFGS (L-BFGS), conjugate gradient (CG) and adaptive moment estimation (Adam) [48], [49], [50], [51], [52].

4.2.2 Recurrent Neural Networks and Long Short-Term Memory

In a simple RNN architecture, input features are updated each instant t and fed into an ordinary ANN, where hidden layers are connected to form a feedback path. Then, the

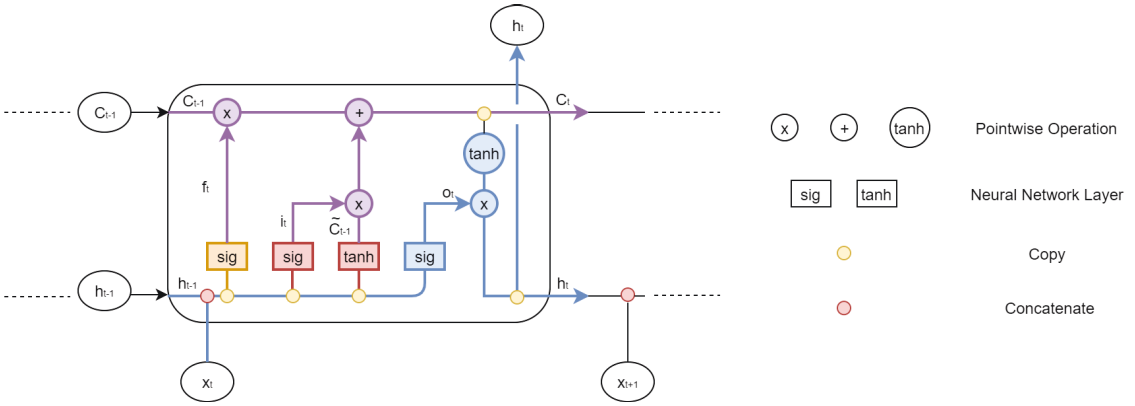


Figure 8: Long short-term memory architecture.

serving BS collects past instants of the MTDs to use as the training set, where a certain model with proper weights should describe the sequence of this training set and how each instant is related to the previous ones. Using RNN, we can extract this pattern and form the correct weights, which will be used to predict new future instants. In such RNNs, there are three types of weights: *i)* w_{xh} represents weights from input features to hidden layers; *ii)* w_{hh} corresponds to weights from hidden layers at time instant t-1 to hidden layers at time instant t; and *iii)* w_{hy} yields the weights from hidden layers to output. Those weights are used in the underlying prediction as follows:

$$h_t = \tanh(w_{hh}h_{t-1} + w_{xh}x_t), \tag{28}$$

$$y_t = w_{hy}h_t, \tag{29}$$

where $\tanh(\cdot)$ is the hyperbolic tangent activation function, h_{t-1} is the previous hidden layer at $t-1$ result from the same recurrent equation, and x_t is the input features vector at t. Afterwards, a loss function and an optimizer are applied to find the correct weights that describe the relation between inputs and outputs [49].

Despite their astonishing ability to forecast future, the RNNs still have a major weakness, namely they have difficulty to extract relevant information located in the far past. Long sequences cause a major problem known as vanish gradient, where the relevant information is located far away in the past experiences almost zero gradient [53]. To solve the problem of long-term dependencies, Hochreiter and Schmidhuber introduced their LSTM architecture in 1997 [28]. It uses the same concept of basic RNN, but with complex four-gate functions connecting past and current instants to extract relevant information from long and short memories. As shown in Fig. 8, the LSTM has two inputs at each instant the short term and the long term memories given by h_{t-1} and C_{t-1}, respectively. While the former yields the previous hidden layer just as in RNN, the later allows the LSTM to learn what to add and what to ignore from the very long past to keep only relevant information for prediction. The LSTM can be classified into 4 main gates:

1. The forget gate describes accurately what information to forget from both input features and previous hidden states by running an ANN and updating its weights, W_f, and biases, b_f. The cprresponding output, f_t, is a number in $(-1, 1)$ as it

results from a sigmoid function σ. Here, -1 means completely forget a feature, while $+1$ means completely keep it. Forget gate result is calculated by:

$$f_t = \sigma(W_f \left[h_{t-1}, x_t\right] + b_f). \tag{30}$$

2. The learn gate is similar to ordinary RNN, as it is responsible for learning new features related to the model. It runs an ANN and updates its weights, W_i, and biases, b_i, to form the vector i_t. Moreover, it creates a new vector, $\tilde{C}_t$, with possible features that can be added afterwards. Learn gate equations are given by:

$$i_t = \sigma(W_i \left[h_{t-1}, x_t\right] + b_i), \tag{31}$$
$$\tilde{C}_t = \tanh\left(W_C \left[h_{t-1}, x_t\right] + b_C\right). \tag{32}$$

3. Remember gate form the long term memory of this time instant, C_t, using the results of the forget gate and the learn gate. Long term memory is calculated as:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t. \tag{33}$$

4. Use gate is the last part of LSTM architecture and aims to update the current short term memory as the current hidden layer and to be used in prediction as the previous hidden layer in the next iteration. It runs an ANN and updates its weights, W_o, and biases, b_o, to form the vector o_t. Use gate equations are given by:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o), \tag{34}$$
$$h_t = o_t \odot \tanh(C_t). \tag{35}$$

The serving BS needs to collect a relevant dataset, which contains the status of each MTD for a period of time. This dataset is then trained with an LSTM model to predict the status of the MTDs in the future for a certain period of time. The training and prediction phases should be continuously alternated. We divide the time axis into windows, each window consists of two phases: *i)* training phase, and *ii)* prediction and correction phase. In the training phase, the BS uses the collected dataset about each MTD activity for a certain period of time T_{train} to use it with the LSTM model to predict its activity in the following $T_{predict}$. The prediction is subject to a degree of accuracy and therefore prone to errors. Particularly in this context, there are two types of errors:

1. the device is silent and it is predicted as active;

2. the device is active and it is predicted as silent.

Assume the BS has feedback, which we will discuss later, that senses these errors and corrects them. Define ΔT as the correction time. The serving BS should correct errors, that exist in the first $T_{predict} - \Delta T$ period of the predicted samples. Afterwards, a new training phase starts by shifting the training window by $T_{predict} - \Delta T$, where the BS uses the corrected prediction samples concatenated with the last $T_{train} - (T_{predict} - \Delta T)$ period of the previous training period to train new samples using the same model. Then, a prediction will be performed for the next $T_{Predict}$ with correction of errors for first the $T_{predict} - \Delta T$ interval of predicted samples. The process repeats so that we have error-free data at every training phase. The proposed algorithm is illustrated in Fig. 9.

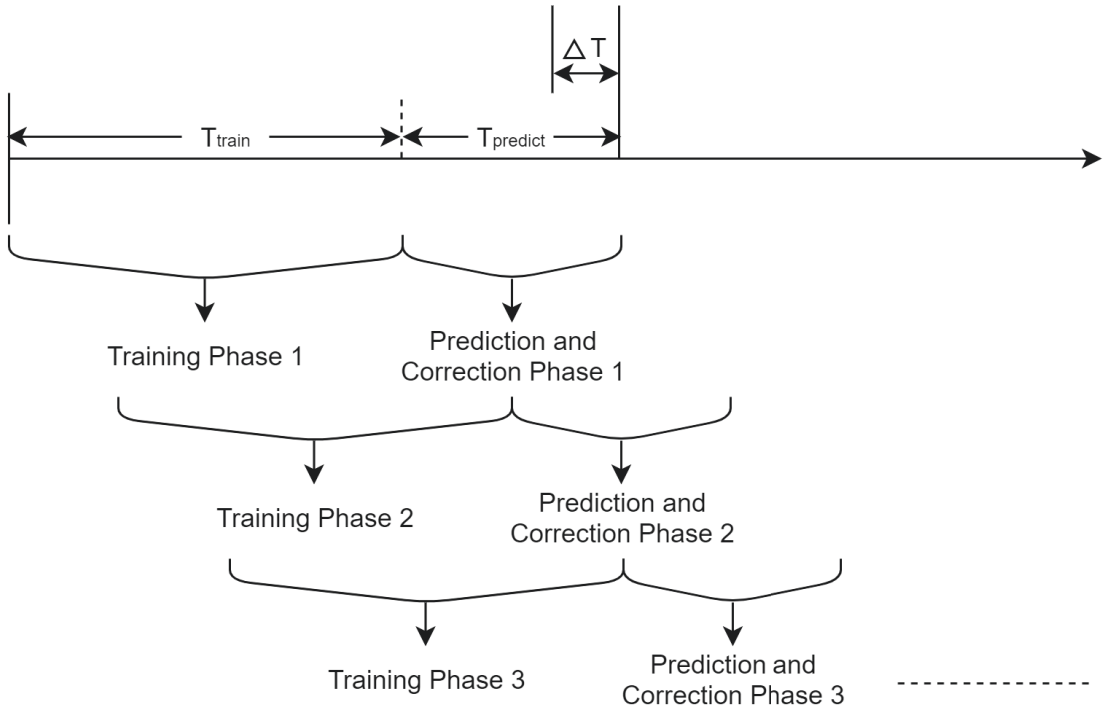


Figure 9: Visualizing prediction of MTD status in real-time. Time axis is divided into windows, which are composed of training and prediction phases.

4.3 Resource Allocation

As the serving BS classifies the type of each MTD and predicts their traffic, the next step is to schedule the resources for these devices. The resource allocation algorithm is illustrated in Fig. 10. The BS predicts the active devices at each instant, then classifies their priorities and grants them the needed resources with given order according to their priorities. In addition, the BS performs some error correction techniques to perform an accurate prediction without accumulation of errors at each phase. The serving BS implements different procedures to carry out feedback and error correction,

1. the serving BS avoids eventual prediction errors by adding safety margins. The length of T_m is chosen according to an exploration rate[1] α.

2. When the MTD is predicted to be active, while it is silent, the BS senses that the allocated resource has not been used. Afterwards, the status of this device is changed from active to silent in the dataset to avoid wrong future prediction.

3. When the MTD is predicted to be silent, while it is active, the device should wait a period of time T_m to get a resource. If it does not get a resource, it should communicate with the BS using RA procedure. Hence, the BS should dedicate some resources for RA procedure to be used in case of errors in prediction. The

[1]The exploration rate α controls the margin time T_m, available resources for RA, and available resources for random allocation. Adjusting different exploration rate is out of scope this work, therefore, we arbitrarily set these parameters based on experimentation.

percentage a of dedicated RA resources is also adjusted according to the exploration rate α.

4. The BS explores random devices, other than those that it has predicted, using available unused resources randomly according to α.

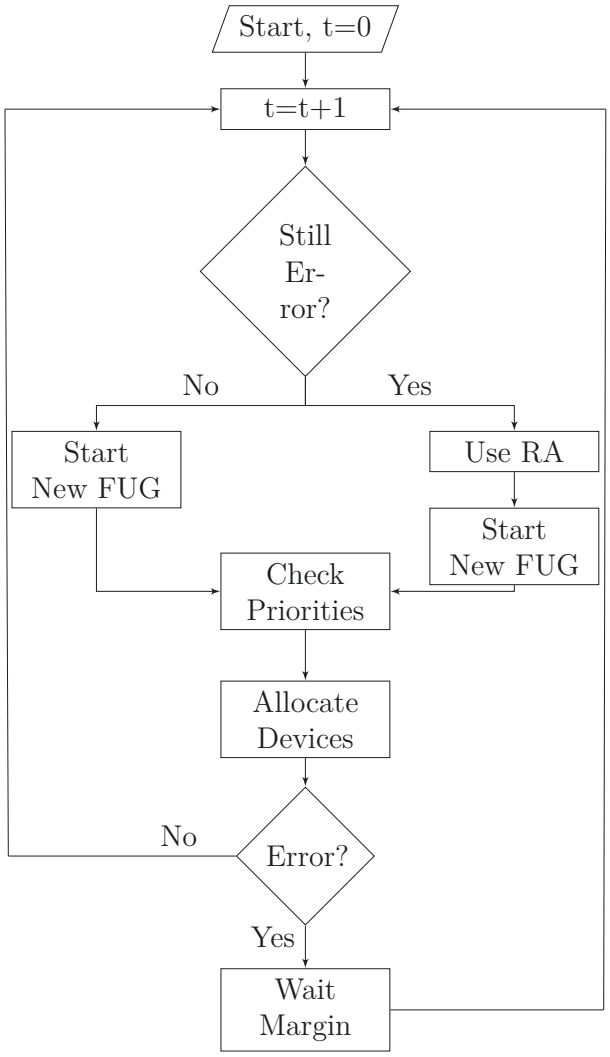


Figure 10: Fast Uplink Grant resource allocation algorithm.

5 SIMULATION RESULTS

In this section, we present the simulation results of the proposed FUG. First, we introduce the performance evaluation metrics. Afterwards, the SVM classification results are presented and compared to other classifiers. Then, we present the LSTM prediction of sensors activity. Finally, we apply those results along with the error correction techniques to schedule resources to MTDs. The proposed FUG model is compared with GB-RA, random FUG, where the serving BS randomly allocates resources to devices, and genie-aided FUG, where BS knows perfectly the traffic of each device and its priority. This comparison is done while adjusting the number of available frequency resources dedicated for 1000 devices in a 1000 m $\times$ 1000 m deployment area within 60 seconds. Moreover, The exploration rate α is set to 0.1, a $= \alpha$, and $T_m = 20\,ms$.

5.1 Performance Evaluation Metrics

Both the device classification and traffic prediction algorithms are considered as binary classification problems. Thus, there are many appropriate evaluation metrics, which are suitable for rare alarms applications such as [54], [55]:

- **The Confusion Matrix** is the most important evaluation method, which illustrates the number of correct and wrong classifications in each class.
- **The Classification Accuracy** describes the overall performance of the classifier.
- **The Precision and Recall.** The former describes the ratio of true predicted samples for each class to the total predicted samples of that class, while the later describes the ratio of true predicted samples for each class to the total actual samples of that class.
- **The f1-Score** ($f1s$) combines the P and R measurements via harmonic mean, resulting in a percentage near to the minimum between them.

$$f1s = \frac{2\,R\,P}{R+P}. \tag{36}$$

We evaluate the performance of the network with respect to throughput. The Transmission rate is calculated using $C = \log(1 + \mathsf{SNR}\,|h|^2)$, where SNR is the signal to noise ratio, and h is the given channel coefficient between an MTD and the serving BS. Each MTD will have a certain rate depending on the SNR and the channel condition between that device and the BS. In addition, for each transmission, different rates exist depending on the transmission power of an MTD and channel condition at time of transmission. As our main scope is to compare FUG to other allocation techniques, we assume the radio channel to be degraded by flat fading.

Communication systems have different sources of delay such as hardware delay T_h, queuing delay T_q, and transmission delay T_t. The access delay, T_a, is the time difference between the moment an MTD is ready for transmission and the moment it gets a resource. It is a function of hardware delay, signaling overhead delay, and queuing delay originated from the existence of lower number of resource compared to the number of ready devices

at a time. In our simulation, we neglect the hardware delay and focus only on the access delay with respect to queuing delay and message exchange between MTDs and the BS. The access delay can be calculated as follows:

$$T_{total} = T_h + T_{overhead} + T_q + T_t, \tag{37}$$

$$T_a = T_h + T_{overhead} + T_q = T_{total} - T_t. \tag{38}$$

5.2 Simulation Results

5.2.1 Device Classification

We initially collect the training set by running multiple CMMPP and M-CMMPP simulations with uniformly distributed epicenters, variances, and intervals of background processes. Then, we pre-process the data by balancing, normalizing and then removing redundancies to ease the training phase and extract the correct features. Afterwards, we compare different classification algorithms to a new CMMPP/M-CMMPP model as shown in table 2. We observe that a polynomial kernel (degree = 6) SVM provides good performance in CMMPP case, but fails in the network congestion case. An ANN architecture with 4 hidden layers (16, 32, 8, and 4 neurons) works very well in both cases in terms of data and alarm classification, where it introduced the lowest errors (only 12 alarm errors in case of CMMPP and 2 alarm errors in case of M-CMMPP) in classifying alarms compared to all classifiers. However, it has relatively large number of errors classifying data devices. The radial basis function SVM, decision trees (DT), and random forests (RF) produce better results than ANN in classifying data devices and almost similar results as ANN in classifying alarm devices. In addition, they provide precise figures. overall accuracy. Radial basis function SVM is the simplest algorithm among them and works extremely fast. It produces a recall of 0.87 and 0.88 for data and alarm classification, respectively in case of CMMPP traffic and a recall of 0.96 and 0.97 for data and alarm classification in case of M-CMMPP traffic. These small number of errors reflect the strength of the rbf SVM classifier and introduces low amount of errors will prioritizing the devices. Hence, rbf SVM is the chosen algorithm to be used by the BS to classify devices.

5.2.2 Time Prediction

The Numenta Anomaly Benchmark (NAB) is a time-series dataset, which contains 58 time-series data files designed to help researchers in time-series prediction and streaming anomaly detection applications [56]. This dataset provides real-time data collected from sensors monitoring different physical quantities in industry deployment scenarios. We do some pre-processing steps, where the sensors are active and need resources when its measurement exceeds a certain threshold. After pre-processing steps, 2 months training data have been prepared to be used for 10 minutes prediction data.

We build up an LSTM architecture with 2 hidden layers (150 and 100 Neurons), 20% dropout, mean square error (MSE) loss function, 50 unrolling (create array of 50 samples, then, for the next array, add one element and use the last 49 from the previous array),

Table 2: Confusion matrix, accuracy, precision & recall and f1-score for CMMPP device classification (support: 544 data and 113 alarm) and M-background processes CMMPP device classification (support: 539 data and 153 alarm).

	Algorithm	Conf. Mat.	Acc.	$P\&R$	$f1-Sc.$
CMMPP	Poly. SVM	354 190 7 106	0.90	0.98 0.65 0.36 0.94	0.78 0.52
	rbf SVM	474 70 14 99	0.87	0.97 **0.87** 0.59 **0.88**	0.92 0.70
	Sig. SVM	245 299 64 49	0.45	0.79 0.45 0.14 0.43	0.57 0.21
	ANN	462 82 12 101	0.86	0.97 0.85 0.55 0.89	0.91 0.68
	DT	478 66 17 96	0.87	0.97 0.88 0.59 0.85	0.92 0.70
	RF	473 71 15 98	0.87	0.97 0.87 0.58 0.87	0.92 0.70
M-CMMPP	Poly. SVM	539 0 153 0	0.85	0.78 1.00 0.00 0.00	0.88 0.00
	rbf SVM	518 21 5 148	0.97	0.99 **0.96** 0.88 **0.97**	0.97 0.92
	Sig. SVM	539 0 153 0	0.78	0.78 1.00 0.00 0.00	0.88 0.00
	ANN	486 53 2 151	0.92	1.00 0.90 0.74 1.00	0.95 0.85
	DT	518 21 8 145	0.96	0.98 0.96 0.87 0.95	0.97 0.91
	RF	517 22 6 147	0.96	0.99 0.96 0.87 0.96	0.97 0.91

50 epochs, and using the Adam optimizer [48]. In our predictions, this architecture produces overall accuracy of 95%, the predictor failed 11 times to correctly infer the sensor activity of total 171 activation instants, and it wrongly predicts 41 times that the sensor is active, while it is silent. Furthermore, it achieves f1-Scores of 0.98 for silent prediction and around 0.90 for active predictions. We should point out that much deeper LSTM architecture would enhance the prediction accuracy as it may extract more relvenat information, but it would increase complexity and time of training and prediction as well. In addition, the length of the training phase can be increased, at expense of longer time to extract the pattern. Selecting the appropriate neural network architecture is a very challenging research problem, where large architectures consume long time, whereas short ones may introduce lower accuracy [57]. The depth of the model and the length of the training data should be adjusted according to the available hardware at the BS, criticality of the application, number of devices, and the amount of expected errors. The resulting errors are corrected, as mentioned, to have a clean training data again to be used in the next phase.

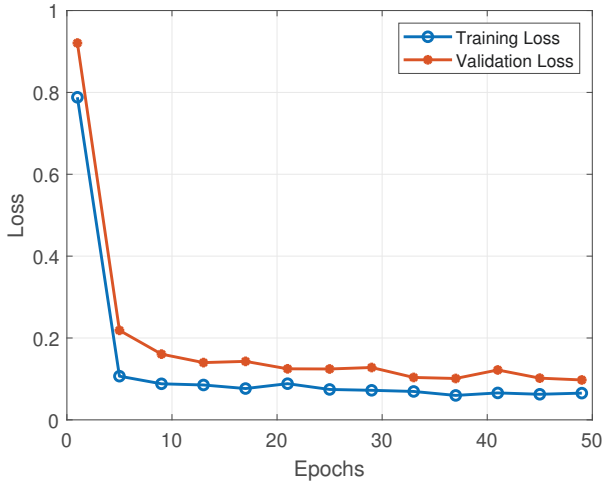
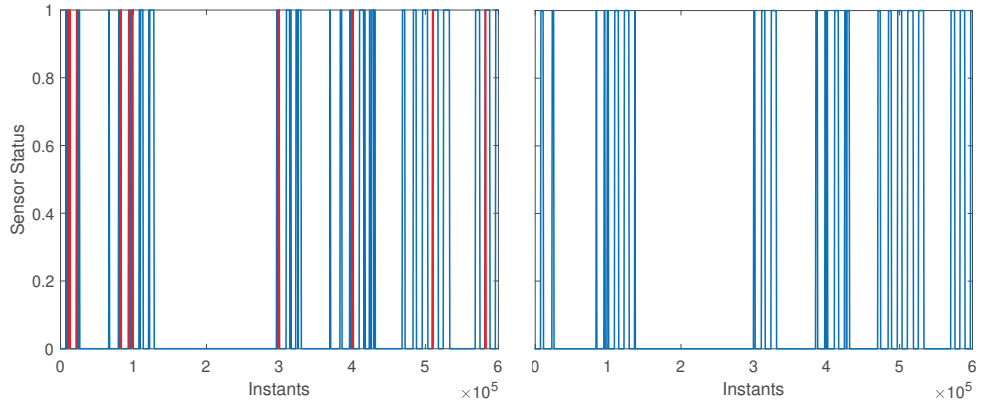


Figure 11: Training and validation losses of sensor activity prediction using LSTM.



(a) Actual Sensor Activity (b) Predicted Sensor Activity

Figure 12: Sensor activity prediction using LSTM. Two months of training data, ten minutes of prediction data and two hidden layers (150 and 100 Neurons).

5.2.3 Resource Allocation

Throughput and access delay are monitored for different allocation schemes while adjusting the number of available frequency resources at the BS. In Fig 13, the throughput, which is the total successfully received packets, is plotted while adjusting the number of available resources during 60 seconds. Random FUG resource allocation has the worst performance, whereas the predicted FUG outperforms the GB-RA and almost achieves the genie-aided FUG for both CMMPP and M-CMMPP. We notice that as the number of available resources increase beyond 75 frequency resources, all schemes perform well.

In Figs. 14 and 15, the average access delay and maximum access delay are plotted, respectively. The proposed predicted FUG almost approaches the genie-aided FUG. Furthermore, the proposed FUG presents free-collision resource allocation scheme, whereas GB-RA suffers from several preamble collisions. The random FUG is presented to mention the importance of having a traffic predictor with high accuracy. Having low traffic

prediction accuracy can cause even worse performance than the RA schemes. We notice that the predicted FUG algorithm achieves extremely low latency in the order of 1 to few milliseconds and high reliability results, where at least, total of 120 Gbytes of packets are successfullt received by the BS.

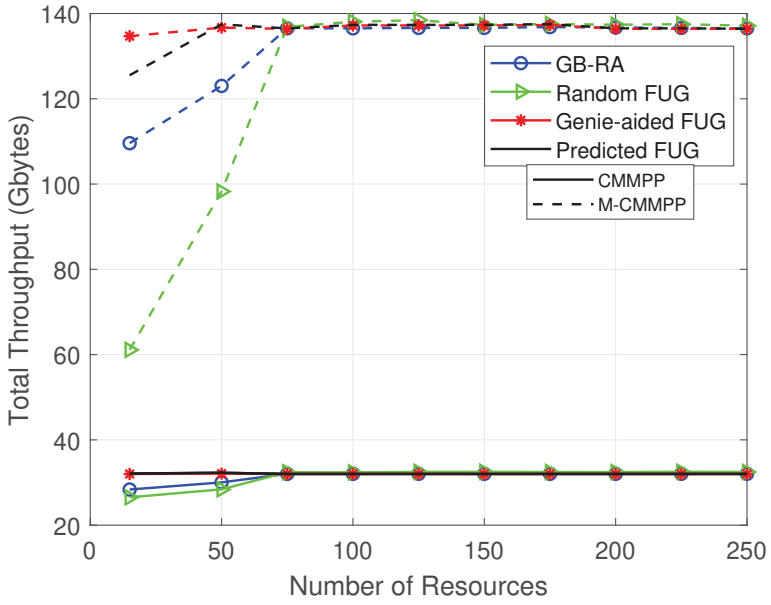


Figure 13: Total throughput while adjusting the number of available resources: 60 s runtime, 1000 MTDs, $\alpha = 0.1$, $a = 0.1$, and $T_m = 20\,ms$. Beyond 75 frequency resources, all schemes converge.

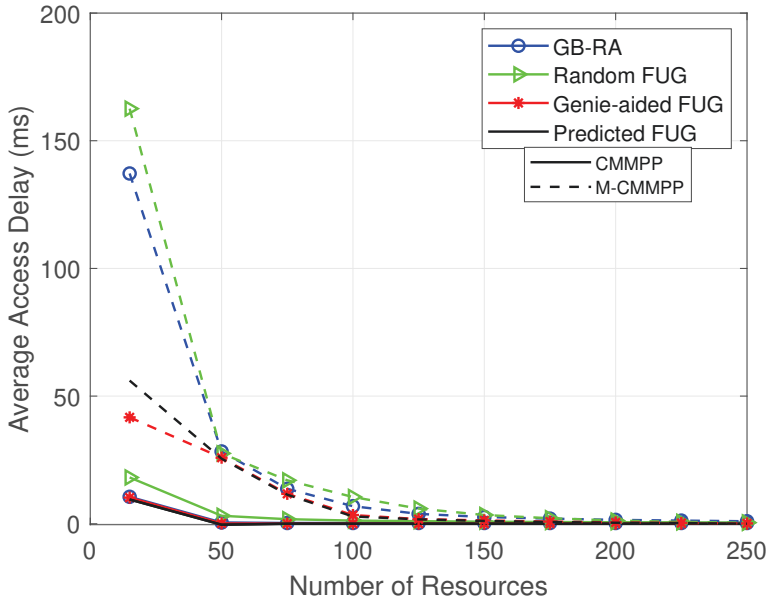


Figure 14: Average access delay while adjusting the number of available resources: 1000 MTDs, $\alpha = 0.1$, $a = 0.1$, and $T_m = 20\,ms$. Beyond 75 frequency resources, all schemes converge.

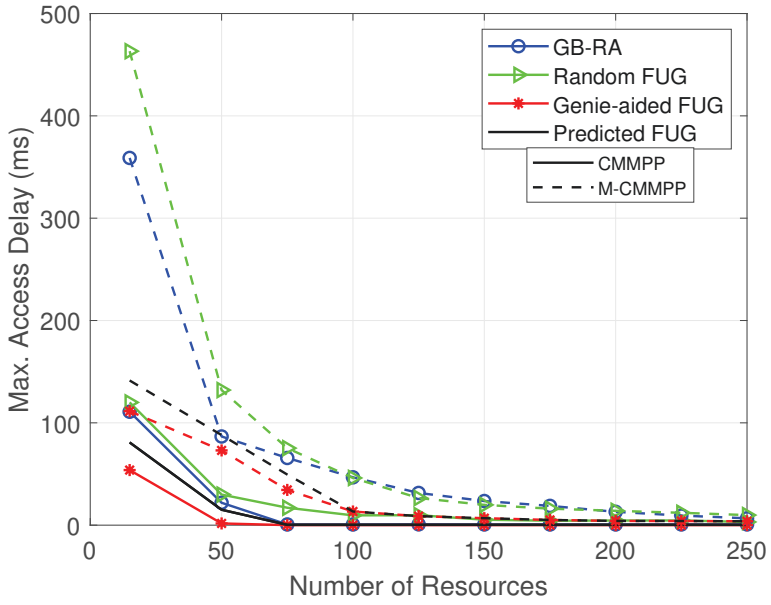


Figure 15: Maximum access delay, while adjusting the number of available resources: 1000 MTDs, $\alpha = 0.1$, $a = 0.1$, and $T_m = 20\,ms$. Beyond 75 frequency resources, all schemes converge.

6 DISCUSSION

This thesis focus on implementing an efficient learning-based resource allocation scheme to grant access to MTDs. The proposed RA allocation techniques are very efficient serving HTC in terms of latency and reliability. They use the 4-handshake RA procedures to establish a communication between the BS and a device. This procedure results in high signaling overhead delays, which fail to achieve the MTC applications demands. The grant-free solutions solve the signaling overhead problem at the cost of higher collision rate. There are many RA based solutions in the literature [17]. However, the traditional approaches suffer from the undesired signaling overhead or collision problems. The 3GPP introduced new features such as learning-based solutions, which allow for implementing fast uplink grant allocation schemes.

Traffic models play an important role in evaluating different allocation schemes. CMMPP traffic model is very efficient and suitable for MTC applications, where it overcomes the complexity of source traffic models and the low efficiency of aggregated traffic models. The M-background processes CMMPP represents a dense scenario for comparing different schemes in a congested network. MTDs have different QoS requirements, which result in distinct priorities for each MTD. SVM classifiers are very efficient and simple algorithm, which can be used in classification in supervised machine learning problems. To design the FUG allocation scheme, we need an efficient traffic predictor to forecast the activity of each MTD. A less-efficient traffic predictor can cause random allocation, which increases the latency and wastes the allocated resources to silent devices. The serving BS uses the results of the device classification using SVM and traffic predictor using LSTM to schedule the resources for the devices. The results of this thesis showed significant results in terms of throughput and access delay, which enables the possibility of serving extremely critical applications like remote surgery and industrial IoT. This work can be extended as follows:

1. Adding Rayleigh fading to the simulation, where we need to evaluate the robust of the model in different channel conditions.

2. Reinforcement Learning (RL) solutions, as in [22], can be combined with the proposed model to achieve the optimum scheduling in applications that have large variety of priorities among their devices or dynamic QoS requirements.

3. Applying multi-connectivity techniques [58], [59] along with FUG can boost the performance of the scheduler and achieve even lower latency and higher reliability.

4. Choosing the appropriate exploration rate α dynamically using learning schemes based on the availability of resources, network congestion, criticality of MTC application, and previous records of prediction accuracy.

7 SUMMARY

In this thesis, we propose a novel fast uplink grant resource allocation scheme based on SVM and LSTM. First, we set up the CMMPP and M-background processes CMMPP models as our system model. Next, we implement an SVM algorithm to classify devices into different priorities. Then, we develop an LSTM architecture to predict traffic and estimate activation time of each MTD. Afterwards, device classification and traffic prediction are used to schedule the resources. Our simulation results show that the proposed FUG outperforms RA schemes and almost approaches genie-aided FUG in terms of throughput and latency. We also present different kinds of error correction techniques that should be used to avoid error accumulation. Equally important, we also show the importance of having an accurate traffic predictor to avoid random allocation behavior.

8 REFERENCES

[1] et al. N.H.M. (2020), White paper on critical and massive machine type communication towards 6G.

[2] Mahmood N.H., Alves H., López O.A., Shehab M., Osorio D.P.M. & Latva-aho M. (2019) Six key enablers for machine type communication in 6G. 6G Summit .

[3] Xu L.D., He W. & Li S. (2014) Internet of things in industries: A survey. IEEE Transactions on Industrial Informatics 10, pp. 2233–2243.

[4] Adas A. (1997) Traffic models in broadband networks. IEEE Communications Magazine 35, pp. 82–89.

[5] Nikaein N., Laner M., Zhou K., Svoboda P., Drajic D., Popovic M. & Krco S. (2013) Simple traffic modeling framework for machine type communication. In: ISWCS 2013; The Tenth International Symposium on Wireless Communication Systems, pp. 1–5.

[6] Centenaro M. & Vangelista L. (2015) A study on M2M traffic and its impact on cellular networks. In: 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), pp. 154–159.

[7] Laner M., Svoboda P., Nikaein N. & Rupp M. (2013) Traffic models for machine type communications. In: ISWCS 2013; The Tenth International Symposium on Wireless Communication Systems, pp. 1–5.

[8] Grigoreva E., Laurer M., Vilgelm M., Gehrsitz T. & Kellerer W. (2017) Coupled markovian arrival process for automotive machine type communication traffic modeling. In: 2017 IEEE International Conference on Communications (ICC), pp. 1–6.

[9] Dahlman E., Parkvall S. & Sköld J. (2011) 4G: LTE/LTE-advanced for mobile broadband.

[10] Popovski P., Stefanović Č., Nielsen J.J., de Carvalho E., Angjelichinoski M., Trillingsgaard K.F. & Bana A. (2019) Wireless access in ultra-reliable low-latency communication (URLLC). IEEE Transactions on Communications 67, pp. 5783–5801.

[11] Laya A., Alonso L. & Alonso-Zarate J. (2014) Is the random access channel of LTE and LTE-A suitable for M2M communications? a survey of alternatives. IEEE Communications Surveys Tutorials 16, pp. 4–16.

[12] 3GPP (2012) TR 37.868 study on RAN improvements for machine-type communications. Tech. Rep. .

[13] Mahmood N.H., Abreu R., Böhnke R., Schubert M., Berardinelli G. & Jacobsen T.H. (2019) Uplink grant-free access solutions for URLLC services in 5G new radio. In: 2019 16th International Symposium on Wireless Communication Systems (ISWCS), pp. 607–612.

[14] Madueño G., Stefanović Č. & Popovski P. (2014) Reliable reporting for massive M2M communications with periodic resource pooling. Wireless Communications Letters, IEEE 3.

[15] Ali M.S., Hossain E. & Kim D.I. (2017) LTE/LTE-A random access for massive machine-type communications in smart cities. IEEE Communications Magazine 55, pp. 76–83.

[16] Abbas R., Shirvanimoghaddam M., Li Y. & Vucetic B. (2019) A novel analytical framework for massive grant-free NOMA. IEEE Transactions on Communications 67, pp. 2436–2449.

[17] Sharma S.K. & Wang X. (2020) Toward massive machine type communications in ultra-dense cellular iot networks: Current issues and machine learning-assisted solutions. IEEE Communications Surveys Tutorials 22, pp. 426–471.

[18] Li R., Zhao Z., Zhou X., Ding G., Chen Y., Wang Z. & Zhang H. (2017) Intelligent 5G: When cellular networks meet artificial intelligence. IEEE Wireless Communications 24, pp. 175–183.

[19] 3GPP (2015) TS 36.881 study on latency reduction techniques for LTE. Tech. Spec. .

[20] Shehab M., Hagelskjær A.K., Kalør A.E., Popovski P. & Alves H. (2020), Traffic prediction based fast uplink grant for massive IoT.

[21] da Silva M.V., Souza R.D., Alves H. & Abrão T. (2020) A NOMA-based Q-learning random access method for machine type communications. IEEE Wireless Communications Letters 9, pp. 1720–1724.

[22] Ali S., Ferdowsi A., Saad W., Rajatheva N. & Haapola J. (2020) Sleeping multi-armed bandit learning for fast uplink grant allocation in machine type communications. IEEE Transactions on Communications 68, pp. 5072–5086.

[23] Manning C.D., Raghavan P. & Schütze H. (2008) Introduction to Information Retrieval. Cambridge University Press, USA.

[24] Hossin M. & M.N S. (2015) A review on evaluation metrics for data classification evaluations. International Journal of Data Mining & Knowledge Management Process 5, pp. 01–11.

[25] Xie J. & Wang Q. (2018) Benchmark machine learning approaches with classical time series approaches on the blood glucose level prediction challenge.

[26] David E. Rumelhart G.E.H..R.J.W. (1986) R. learning representations by back-propagating errors 323, pp. 533–536.

[27] Zaremba W., Sutskever I. & Vinyals O. (2014) Recurrent neural network regularization. CoRR abs/1409.2329. URL: `http://arxiv.org/abs/1409.2329`.

[28] Hochreiter S. & Schmidhuber J. (1997) Long short-term memory. Neural computation 9, pp. 1735–80.

[29] Cho K., van Merrienboer B., Gulcehre C., Bahdanau D., Bougares F., Schwenk H. & Bengio Y. (2014), Learning phrase representations using RNN encoder-decoder for statistical machine translation.

[30] Heffes H. & Lucantoni D. (1986) A markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance. IEEE Journal on Selected Areas in Communications 4, pp. 856–868.

[31] Miller S.L. & Childers D. (2012) Chapter 9 - markov processes. In: S.L. Miller & D. Childers (eds.) Probability and Random Processes, Academic Press, Boston, second ed., pp. 383 – 428. URL: http://www.sciencedirect.com/science/article/pii/B9780123869814500126.

[32] Niyato D. & Hossain E. (2009) Cognitive radio for next-generation wireless networks: an approach to opportunistic channel selection in IEEE 802.11-based wireless mesh. IEEE Wireless Communications 16, pp. 46–54.

[33] Mitola J. & Maguire G.Q. (1999) Cognitive radio: making software radios more personal. IEEE Personal Communications 6, pp. 13–18.

[34] Mitola J. (2000) Cognitive radio an integrated agent architecture for software defined radio.

[35] Haykin S. (2005) Cognitive radio: Brain-empowered wireless communications. IEEE J. SELECTED AREAS IN COMM 23, pp. 201–220.

[36] Yucek T. & Arslan H. (2009) A survey of spectrum sensing algorithms for cognitive radio applications. IEEE Communications Surveys Tutorials 11, pp. 116–130.

[37] Janatian N., Modarres-Hashemi M. & Sun S. (2015) Sensing-based resource allocation in multi-channel cognitive radio networks. In: 2015 IEEE Symposium on Communications and Vehicular Technology in the Benelux (SCVT), pp. 1–6.

[38] Wu H., Zhu C., La R.J., Liu X. & Zhang Y. (2013) FASA: Accelerated S-ALOHA using access history for event-driven M2M communications. IEEE/ACM Transactions on Networking 21, pp. 1904–1917.

[39] Cheng R., Chen J., Chen D. & Wei C. (2015) Modeling and analysis of an extended access barring algorithm for machine-type communications in LTE-A Networks. IEEE Transactions on Wireless Communications 14, pp. 2956–2968.

[40] Lien S., Liau T., Kao C. & Chen K. (2012) Cooperative access class barring for machine-to-machine communications. IEEE Transactions on Wireless Communications 11, pp. 27–32.

[41] Duan S., Shah-Mansouri V. & Wong V.W.S. (2013) Dynamic access class barring for M2M communications in LTE networks. In: 2013 IEEE Global Communications Conference (GLOBECOM), pp. 4747–4752.

[42] Cheng J., Lee C. & Lin T. (2011) Prioritized random access with dynamic access barring for RAN overload in 3GPP LTE-A networks. In: 2011 IEEE GLOBECOM Workshops (GC Wkshps), pp. 368–372.

[43] Farhadi G. & Ito A. (2013) Group-based signaling and access control for cellular machine-to-machine communication. In: 2013 IEEE 78th Vehicular Technology Conference (VTC Fall), pp. 1–6.

[44] Pratas N.K., Thomsen H., Stefanović Č. & Popovski P. (2012) Code-expanded random access for machine-type communications. In: 2012 IEEE Globecom Workshops, pp. 1681–1686.

[45] Gürsu H.M., Vilgelm M., Kellerer W. & Reisslein M. (2017) Hybrid collision avoidance-tree resolution for M2M random access. IEEE Transactions on Aerospace and Electronic Systems 53, pp. 1974–1987.

[46] Madueño G.C., Stefanović Č. & Popovski P. (2014) Efficient LTE access with collision resolution for massive M2M communications. In: 2014 IEEE Globecom Workshops (GC Wkshps), pp. 1433–1438.

[47] Mahmood N.H., Abreu R., Böhnke R., Schubert M., Berardinelli G. & Jacobsen T.H. (2019), Uplink grant-free random access solutions for URLLC services in 5G new radio.

[48] Kingma D.P. & Ba J. (2017), Adam: A method for stochastic optimization.

[49] Duchi J., Hazan E. & Singer Y. (2011) Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research 12, pp. 2121–2159.

[50] Amari S. (1998) Natural gradient works efficiently in learning. Neural Computation 10, pp. 251–276.

[51] Schaul T., Zhang S. & LeCun Y. (2013), No more pesky learning rates.

[52] Zeiler M.D. (2012), Adadelta: An adaptive learning rate method.

[53] Bengio Y., Simard P. & Frasconi P. (1994) Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks 5, pp. 157–166.

[54] Ng A. (2017) Machine learning yearning. URL: `http://www.mlyearning.org`.

[55] Sokolova M. & Lapalme G. (2009) A systematic analysis of performance measures for classification tasks. Information Processing & Management 45, pp. 427 – 437. URL: `http://www.sciencedirect.com/science/article/pii/S0306457309000259`.

[56] Lavin A. & Ahmad S. (2015) Evaluating real-time anomaly detection algorithms – the numenta anomaly benchmark. In: 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), pp. 38–44.

[57] Hunter D., Yu H., Pukish, III M.S., Kolbusz J. & Wilamowski B.M. (2012) Selection of proper neural network sizes and architectures-a comparative study. IEEE Transactions on Industrial Informatics 8, pp. 228–240.

[58] Mahmood N., Lopez Lechuga M., Laselva D., Pedersen K. & Berardinelli G. (2018) Reliability oriented dual connectivity for URLLC services in 5G new radio. pp. 1–6.

[59] Mahmood N.H. & Alves H. (2019) Dynamic multi-connectivity activation for ultra-reliable and low-latency communication. In: 2019 16th International Symposium on Wireless Communication Systems (ISWCS), pp. 112–116.